UPPSALA
UNIVERSITET

IT 18 046
Examensarbete 15 hp
September 2018

Local Geography Educator

Martin Sellergren

Institutionen för informationsteknologi
Department of Information Technology

UPPSALA
UNIVERSITET

Teknisk- naturvetenskaplig fakultet
UTH-enheten

Besöksadress:
Ångströmlaboratoriet
Lägerhyddsvägen 1
Hus 4, Plan 0

Postadress:
Box 536
751 21 Uppsala

Telefon:
018 – 471 30 03

Telefax:
018 – 471 30 00

Hemsida:
http://www.teknat.uu.se/student

Abstract

Local Geography Educator

Martin Sellergren

The intent of this project is to produce a mobile application capable of teaching local geography of any place in the world to it's users. The meaning of local geography in this context is attributes of villages, city-districts, roads, parks, rivers, schools and such geographic objects – object inside a limited area. The goal was to make a stimulating, entertaining and above all educational application. A design was carefully assembled through an analysis of existing applications with a similar concept, through a review of human factors regarding memory and learning, through an interview with a potential user, among other things. The corresponding application was implemented for Android mobile phones and tablets. The application introduces quiz-based education with well-reasoned solutions to design difficulties also faced by other developers of applications of this kind. The resulting application is likely unique with the functionality to offer education of detailed local geography of any place in the world. A brief evaluation of the educational value of the resulting application had promising results.

Handledare: Dave Clarke
Ämnesgranskare: Iordanis Kavathatzopoulos
Examinator: Olle Gällmo
IT 18 046
Tryckt av: Reprocentralen ITC

Contents

1 Introduction

Since recent years there has aroused a new popular way of education, through the mobile platform. Many mobile applications exist with an educational purpose, and some has grown very popular. It turns out the mobile platform is well suited for these purposes. A mobile phone is easily accessible as it's often carried around, and it's easy for the owner to do a little studying whenever there is a desire and some time to spare. Moreover, mobile application developers have good potential to make an application stimulating and entertaining using this platform. [1]

The intent of this project is to produce a mobile application capable of teaching local geography to its users. The meaning of local geography in this context is attributes of villages, city-districts, roads, parks, rivers, schools and such geographic objects – object inside a limited area, like a city. The goal is to make a stimulating, entertaining and above all educational mobile application. This application will be referred to as *Location Educator* in this document.

Location Educator can for example be useful for someone who wants to gain better knowledge of the city where they live or where they want to go to. People who have a desire to be more knowledgeable about some place in the world. There might also be uses in professional areas. People working with transportation or public communication might benefit greatly. For example taxi and bus drivers. Not mainly to find the way to where they are going, which they likely use a GPS for, but more so for being able to answer questions from the customers about directions and distances.

Several mobile applications for geographic education currently exist. Some examples are *Seterra*[2], *World Geography*[3], *Geo Challenge*[4] and *MapPie*[7]. These are all based on education through quizzes. The user is ranked based on the result of a quiz and the goal can be to complete many quizzes with good rankings and progress to new levels. Such a design has potential to provide education through entertainment and will also be integrated into Location Educator. A defining difference with other applications is this project's focus on local geography, where others usually focus on a global or regional scale, such as continents, oceans, states, provinces and cities.

2 Background

To satisfy the project goal of making a stimulating, entertaining and educational mobile application, careful consideration needs to be taken about the application's design. In this section, relevant areas are investigated and steps towards the design decisions are described. Next section describes the derived design.

2.1 Targeted audience

To increase the chance of good human-computer interaction, it may be appropriate to define a targeted audience. The targeted audience of Location Educator is broad, it's defined as anyone interested in learning local geography. Users might be of different ages with different experiences. Therefor, care is taken to make the interface self-explanatory and easy to interact with. Simplicity is

favored over complex functionality. Focus is on inexperienced users over those preferring more advanced functionality. In order to still accommodate many different users, Location Educator will support different speeds in the education progression.

2.2 Educational aspects

It’s important to keep *human factors* in mind. How do we remember and learn? The human learning process is complex. To learn something new isn’t a trivial thing, especially not to learn something and remember it for a long time. [10]

A key to preserved memories is associations since long term memories are stored and retrieved by association. The more associations the better. [10]. If you hear a name of a town you’ve never heard of, it’s easily forgotten. Then, you see it’s location on a map, you here a story about the magnificent mountains there and you meet an inhabitant who turns out to have very red skin, then the name has lot’s of associations, and you’ll more likely remember it. Finally you realize the name sounds a bit like *red mountains* and you’ll never forget it.

The mind forgets and needs to be reminded. Therefore is repetitions important for the learning process. The longer the knowledge has existed, the less often a reminder is needed. [11]

Stimulation is important. A healthy amount of stress will facilitate the learning process. Too little stress and the mind might start thinking of other things. Too much and it will be disturbing and desired to escape it. [12]

It’s likely beneficial for the person being educated to realize that errors and mistakes are okay – it’s a part of the learning process. At the time of a mistake, don’t say “Wrong -next question!”. Instead, take this opportunity to make it right, explain why it’s wrong and let the person try again, perhaps this time with a clue.

2.3 Initial potential user considerations

An interview with a potential user was performed before any major design decisions were made. The person was shown some initial sketches of a design. Through discussion, it became clear that a light workload for the user is of importance. Initially, the plan was to let a user study a particular area for a considerable amount of time and then perform an exam for this area in order to move on to next area and level. After the interview this idea was modified into a design that doesn’t involve the risk of users getting stuck in the exercise-progression as easily. The new design was discussed with the potential user who agreed that this was an improvement. The modified design involved lightweight levels with subsequent reminders (more details later in the report).

2.4 Existing application analysis

There currently exist multiple geography education applications. They are often based on learning through quizzes. The user answers questions like "Where is Stockholm located?", or "What city is this?" and answers through tapping in a map or picking one from multiple alternatives. But the applications differ greatly in key aspects like how to stimulate motivation and goal. Some of these

applications have been studied in detail in order to identified strengths and weaknesses from an educational point of view. The results from this study is described in next subsection. Focus of this study was on the following applications: *Seterra*[2], *World Geography*[3], *Geo Sverige*[4], *Geo Challenge*[5], *Countries of the world*[6], *MapPie*[7] and *Blank Map Quiz*[8].

Worth pointing out is that these applications all have a focus on global or regional geography, that is education of things like continents, oceans, countries, capitals and important landmarks. Despite this difference in focus with Location Educator, they can offer valuable insights in how to, and not to, design an educational geography application.

Furthermore, the language education application *Duolingo*[9] was studied. This application is very popular, which I believe is mainly because of an effective education design. Duolingo also supports quiz-based education and has provided inspiration and ideas.

2.5 Key-functionality for education

Through the existing application analysis and other considerations, some key-functionality-aspects regarding educational value in a quiz-based educational application has been suggested. These aspects are described in the subsections below, along with how they are handled in applications in the existing applications study. Considerations about how successful these applications have been in these areas was used as foundation for design decision of Location Educator.

2.5.1 Motivation

What makes the user want to continue using the application? Probably because of a desire to learn local geography, but there may be a need of a specific motivator that indicates that progress are made. One way to implement this is a hunt for high-score – the user completes a quiz and gets a score, then retakes the quiz to improve the score. The motivator is to achieve good scores on lot's of quizzes. Another motivator is level progression – unlock levels by getting good results in levels currently unlocked, and try to unlock everything (at which point the user is a master in this area). Some of the studied applications use a high-score based system, some use level progression and some use both.

An example of an application that is high-score based is *Seterra*, where lot's of quizzes are predefined (that is, the content of the quiz doesn't change next time it’s taken). The user can freely select interesting quizzes and tries to get a good high-score on those. This gives the user great freedom to select what to learn, and it's likely a powerful educational strategy to retake quizzes and improve the score. A user may take comfort in being able to come back to the same quiz that has been taken before. A possible downside is that it may seem daunting to some users to select a quiz from the big selection. And also, there is a lack of overall progression which otherwise could provide the user with the sense of going somewhere. The high level of freedom is good in many ways but also means that the user must be disciplined in selecting relevant quizzes and retake them before they are forgotten.

An example of an application that uses level progression is *World Geography*. Here, a quiz is automatically and semi-randomly generated (from some specifications from the user) and by

completing a quiz the user gains points that eventually leads to next level. A new level means an increase in difficulty and more options for quiz-specification. This system is good for motivation since there is a global progression (reaching new levels), and it opens up for a guided increase in difficulty.

The conclusion is that a level-based system is necessary to provide guided progress (like increasing difficulty), and to avoid user feeling daunted when selecting a quiz. Therefor, a level-based system will be implemented in Location Educator. To battle the potential restricted freedom of choice this might involve, Location Educator will group exercise content into categories (more on this later).

2.5.2 Question design

The difficulty of a question should ideally be related to the user's knowledge so that it becomes challenging to answer and not too difficult or easy. If a user only has a slight idea of a subject, it's very rewarding with easy questions that capture this little knowledge and strengthen it. [13]

Question asking and answering should be swift and simple. Questions may vary in difficulty, and it's important for initial questions to be easy to answer. It's likely a good idea to (to some extent) guide the user to the correct answer so the user leaves the question feeling strengthened and not confused.

Multiple choice questions are good in this regard. "What is the name of the road?", a map with a highlighted road is presented along with some answer alternatives, where the user picks one. Answering is swift and simple, and the user is gently guided to the correct answer since there is a limited number of of alternatives to pick from. The user may also be exposed to names of other geographic objects which never is a bad thing. This is a very popular question type in the studied applications.

Answer through typing is not nearly as common, but exists for example in the application *Geo Sverige*. It's often quite tedious, but opens up for a new level of difficulty which sometimes may be preferred.

Another question type is *pin-on-map*-questions where the user answers by tapping somewhere in a map. Two types exist of this, either tapping a blank map, or tapping a map with predefined answer points or areas. Tapping a blank map means the user will have to tap close enough to the exact answer. *Geo Challenge* is an example of an application using this. A problem is that the user will never be able to tap exactly correctly so I believe such questions will have a tendency to leave users feeling unsatisfied. Tapping predefined points or areas may often be preferable, which also has the potential to guide the user in correct direction because of the limited number of answer alternatives. *Seterra* is an application which relies solely on this type of question.

As mentioned in the Educational aspects section, it’s important with multiple associations to support the memory during the learning process. Therefor it may be important with many different types of questions, to attack a subject from different angles. The notion that multiple associations support the memory and learning is beautifully utilized by the application *Countries of the world*, where questions are presented not solely through text or map. In addition, flags and capitals are included.

For Location Educator, multiple types of questions will be implemented, including multi-choice and pin-on-map questions. It is also beneficial to introducing additional associations, but such a design isn't very straightforward in this context. A local geographic object will have a name, a shape and a category. In order to add additional associations, each geographic object will also have a color that is used when the object is presented, which is loosely based on the characteristics of the object. This additional potential association might occasionally offer educational support.

2.5.3 Educational support after incorrect answer

What should happen when the user answers a question incorrectly? This may be of great importance when it comes to educational value, as mentioned in Educational aspects. An incorrect answer is a great opportunity to offer educational support.

Some of the studied applications offers no support, instead moving on to next question, leaving the user feeling confused. Most commonly, applications give the user multiple attempts with a question. The number of attempts varies, often three, sometimes more. Often, I would prefer less number of attempts. If the first or second answer is wrong it's likely that I don't know the answer. Then I would prefer to be given the answer and move on instead of keep guessing. *Seterra* gives the user three attempt, then reveals the answer and waits for the user to click the correct answer. I find it pleasant to always leave a question after clicking the correct answer, even if it was revealed.

It's common to repeat incorrect questions after the quiz; giving the user another chance, which for example is the case in *World Geography* and *Duolingo*.

In Location Educator, when a user answers a question incorrectly, the name or location of the incorrect geographic object will briefly be displayed to the user. Also, the correct answer will be hinted immediately when an incorrect answer is provided, as opposed to other applications which wait for a number of attempts before revealing the correct answer. This ensures a swift feel of the quizzes.

2.5.4 Stimulation

As described in Educational aspects section, stimulation is often a good thing during education. In the studied applications, different ways of implementing stimulation exists. Two main categories in this area exist; a clock ticking down, and a clock ticking up.

In the application *Geo Challenge,* a quiz is assigned some time and the clock is ticking down. The more questions that's correctly answered before time runs out, the better score. This provides lot's of stress, way too much according to me. *World Geography* also has a clock ticking down but it is reset after each question, and the given time is generous. This becomes a mild stimulation which provides a good amount of stress.

A clock ticking up is also common. *Seterra* uses this for example. The time taken to complete a quiz is recorded along with the score, and the goal is to minimize time and maximize score. Through this method, the amount of stress provided through the stimulator is very much dependent on the user. If the user decides to ignore the time, it offers no stimulation at all. Or the user could hunt for new time-records making the stimulator very strong.

Also, some applications don’t use a stimulator at all, for example *Duolingo*.

Because of the potential benefits of a stimulator, one will be implemented in Location Educator. I believe it’s very important that the stimulator isn’t to strong as this would likely make the application uncomfortable to use. Recording the time of a quiz doesn’t seem relevant in this case. Therefore, a counting-down timer will be implemented which is reset after each question, with time generously provided.

2.5.5 Visual appearance

The final suggested key-functionality for education is simply the visual appearance of the quiz. Things like beautiful and contrasting colors, non-cluttered screens and well-thought screen-compositions are very important. Failure here could easily ruin an otherwise good application. This is in some sense the case with the application *MapPie,* where the presented quizzes are arguably very dull in appearance. *Blank Map Quiz* is an example of the opposite. Here, the coloring of the maps presented during quizzes are unsuspected and beautiful.

2.6 Scenario analysis

When making design decisions, it may help to visualize potential users, describe them, their needs and expectations:

- Student in new city, wants to know city better. Busy with school work and activities during daytime but likes to do some light studies of the city before bedtime, for just a couple of minutes. By knowing the city better, the student feels more at home there.
- Elderly person thinking about moving to Portugal in wintertime. Has found a village that seems pleasant and likes to study it in detail and dream about the future. Likes the native names of little bakeries and landmarks. Has been guided into learning the application *Duolingo* and through it refresh the language. Is therefor familiar with the concept of quizzes and progression, even though the person usually finds mobile applications hard to understand.
- A 12-year old who likes to play mobile games and compete with friends. Found Location Educator and likes it since it somehow feels meaningful in a way other games don’t.
- A taxi-driver who doesn’t like to take up his smart phone every time someone asks him for time estimates and directions. Also, he’s embarrassed his boss might find out his really not very knowledgeable about the city. So on his vacation, he focuses flat out on the studies.
- A sailor who plans to explore the archipelago of Stockholm. Wants to learn the names of the little islands since it helps with navigation, and helps preserving his memories of the places he passes.

2.7 Functional requirements

Functional requirements were derived, with support from previous considerations. These are requirements of specific behavior of the application.

- Construct an exercise from a size-limited area anywhere in the world.
- Maintain a list of exercises.
- Group geographic objects inside an exercise area into level-categories based on characteristics.
- Group geographic objects inside a level category into levels based on characteristics.
- Let user take a *level quiz* which takes the user to next level if finished with satisfactory result. The quiz has questions about the geographic objects of this level.
- Let user take a *reminder quiz* – a quiz with questions about geographic objects that have already appeared in a level quiz and are estimated to be weak in the memory of the user. These quizzes are sometimes required in order to open up a level quiz.
- Integrate *follow-up quizzes,* which follows another quiz with incorrect answers from that quiz.
- Include a stimulator (or stressor) during a quiz, through a timer ticking down which is reset after each question.
- Display a quiz progress-bar, as well as overall exercise-progress.
- Support three different types of quiz-questions; *Name-it*, *Place-it* and *Pair-it*.
- Associate each geographic object with a color based on characteristics. This color is used when the object is presented in a map or through text.
- Integrate *exploration*, where a user can practice outside of a quiz.
- Support predefined exercise-areas during exercise creation.
- Exercise-area-sharing and progress-sharing between users.

2.8 Non-functional requirements

Here follows requirements about what the application should *be*.

- The application is required to be educational in local geography. This involves many things, like supporting educational aspects and being graphically and functionally pleasant.
- The application should promote learning about big and important geographic objects before less significant objects.
- Swift, stimulating, lightweight quizzes with basic design, supporting *does what you think*.

- Introduce additional associations when asking questions about a geographic object.
- Educational support during a quiz by guiding user towards correct answer when incorrect answer is given.
- Let user know that he/she is making progress towards a goal.
- Prevent user from forget any previously seen geographic objects. Prefer to remind user of forgotten objects before introducing new ones.

3 Design

Here follows the intended design of Location Educator through descriptions of the different activities of the application. The images are mock-ups produced using image creating software. This design is the goal of the subsequent implementation.

3.1 Select an exercise

The *select exercise* activity displays a list of existing exercises. The user may click on an exercise to enter it, or create a new one. Exercises are reordered through drag and drop, and deleted through a swipe gesture with following confirmation. Menu button top left has various items: About, Feedback, Logout. The mock-up of this activity is presented in Figure 1.

3.2 Create a new exercise

In this activity, the the user can create a new exercise. The user specifies name and area. The area is drawn using draw-tools of the embedded map and can be located anywhere in the world. Only smaller areas are allowed (the application is for *local* geography). This restriction is enforced by disabling drawing tools if map is too zoomed out. If this is the case, the map will appear gray and a button appears that, when clicked, animates the zoom of the map to an allowed level. When zoom level is acceptable, drawing is initiated by tapping the map. As soon as the first node is drawn, zooming and scrolling is disabled. The user may clear the drawn path through a button that appears when the first node is placed. The goal with this design is to enforce the size-restriction of the specified area in a way that is self-explanatory to the user. The mock-up of this activity can be seen in Figure 2.

The functional requirements mentions predefined exercise-areas, which is one way to enable different users to take the same exercise. This may be of interest to users who for instance likes to compare scores with others. The mock-up doesn’t include this feature as this requirements is put aside initially.

3.3 Inside an exercise

When entering an exercise the user is presented with a list of categories, each containing levels, as seen in Figure 3. There are a maximum of five categories, less if the exercise-area doesn’t contain any geographic objects of one or some of the categories. These categories are as follows:

- **Settlements** – geographic objects that may be described as settlements, including cities, towns, villages and neighborhoods. This category also holds things like graveyards, boatyards, and golf courses, basically any kind of named local area that is under human care.
- **Roads** – roads of any sort. Also paths.
- **Nature** – for example forests, nature reserves, islands, beaches, mountains and water bodies of any sort. Also parks of any sort.
- **Transport** – geographic objects related to transportation, for example bus stations, train stations, airports, and ferry terminals.
- **Constructions** – buildings, bridges, tunnels, monuments and so on. Basically any object built by man.

By clicking a category, the user is presented with three options: enter exploration, attempt a level-quiz or take a category-reminder-quiz. These are described in next sections.

Top right is the overall exercise progression. Down right is a button for taking an exercise-reminder-quiz (as opposed to the more restricted category-reminder-quiz). Top left is the menu button which in addition to the items in the menu of the *Select exercise activity* has the option "Select exercise" that takes the user to the Select exercise activity. This options menu may also contain functionality regarding progress and exercise-area sharing.

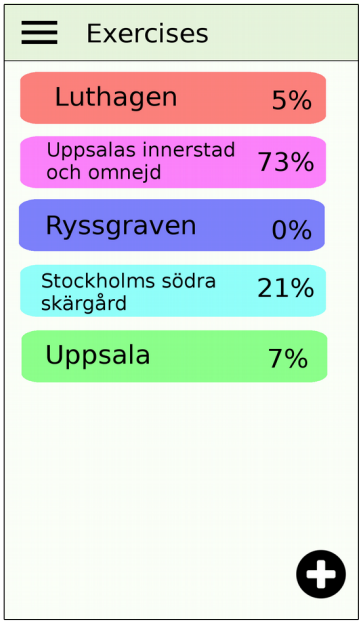

Figure 1: Mock-up of the Select exercise activity.

Figure 2: Mock-up of the New exercise activity.

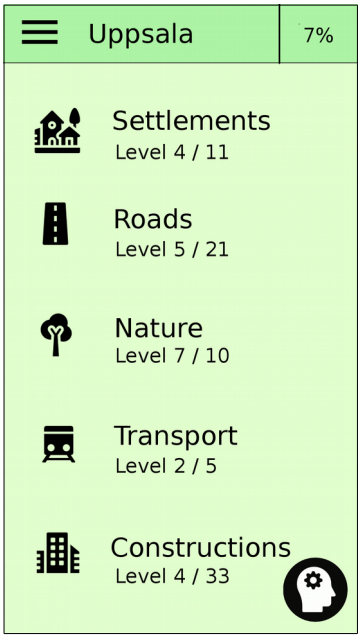

Figure 3: Mock-up of the main activity of an exercise called Uppsala.

3.4 Exploration

Exploration is useful if a user wants to try his/her knowledge outside of a quiz. It's casual practice which may be enjoyed by some users. A similar concept is implemented in the application *Seterra*[2].

The user is presented with a map containing geographic objects of a certain category in an exercise. The user may freely tap objects in order to display their names for a brief period of time. This is particularly useful when a user wants to attempt to name geographic objects and instantly receive confirmation if the guess was correct. A button above the map let's the user decide if to display geographic objects from next level or from all past levels. See the mock-up in Figure 4.

3.5 Quiz

A quiz handles around four geographic objects, with three to five questions each. There are three different types of questions, *Name-it*, *Place-it* and *Pair-it*, described in detail in next sections. A question must be answered within a certain, generously provided amount of time and if no answer is given, the correct answer is shown (this is a stimulator). Each object is associated with a color loosely based on its characteristics, which is used when the object is presented in a question, in order to increase the possible number of association the user has to the object.

There are three different types of quizzes:

- **Level quiz** – for level progression. This quiz has questions about geographic objects of a certain category at a certain level. Increasing levels means less significant objects. The first level may for instance have motorways and the last pathways. Passing a quiz with satisfactory result takes the user to next level of this category.
- **Reminder quiz** – for being reminded of previously seen geographic objects, from previous levels. Problematic and old (long time no see) objects are favored when selecting objects for such a quiz. Reminder quizzes exists both on category-level (for objects in a particular category) and exercise-level (for any seen objects in the exercise). The user is required to take a reminder quiz (or a couple) now and then in order to continue level progression, as seen in Figure 5 and 6.
- **Follow-up quiz** – which follows another quiz (that isn't follow-up) and repeats the incorrectly answered questions.

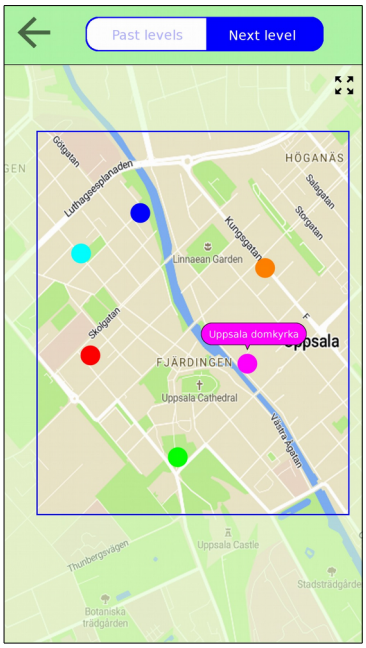

Figure 4: Mock-up of exploration.

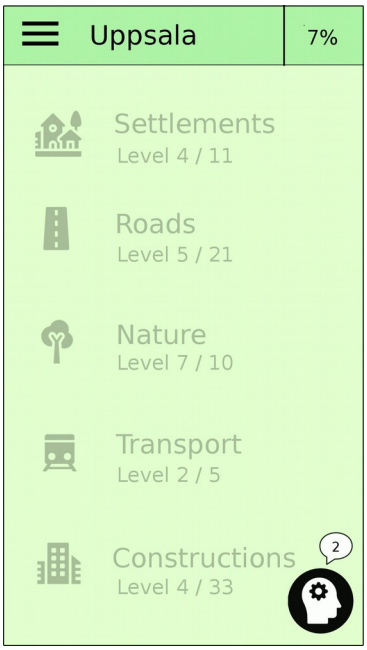

Figure 5: Mock-up of an exercise activity when exercise-reminders are required.

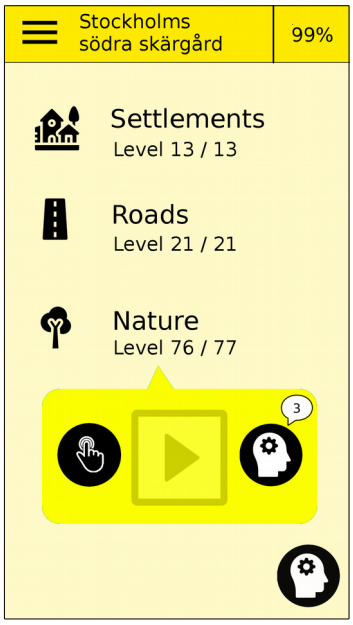

Figure 6: Mock-up of an exercise activity when category-reminders are required.

3.5.1 Quiz question: Name-it

A multiple choice question where a geographic object is presented in a map. If correct answer is given, moves on to next question. If incorrect, the correct answer is hinted and the user chooses the correct alternative before moving on to next question. The number of alternatives is defined by the number of previous questions about this geographic object in the quiz. Initially there are two alternatives, later four and even six. A mock-up is shown in Figure 7.

3.5.2 Quiz question: Place-it

In this question the user is supposed to tap an object presented in a map after being given a name and category. An object may be presented in the map as a single marker, a line or a closed line. Similar concepts regarding answering sequence and alternatives count as for Name-it questions. See the mock-up in Figure 8.

3.5.3 Quiz question: Pair-it

Here the user is supposed to pair name with location, by first tapping a name and then tap the corresponding object in the map. Complete all pairings before the question is done. The geographic objects of this question might not belong to the same level or even category as the quiz. The only requirement is that the objects have been seen in this or earlier quizzes. Therefor, this question-type has the additional function of reminding the user of previously seen objects. The mock-up is presented in Figure 9.

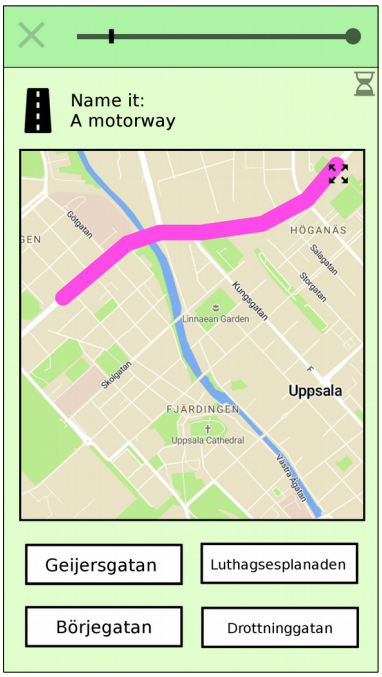

Figure 7: A Name-it question mock-up.

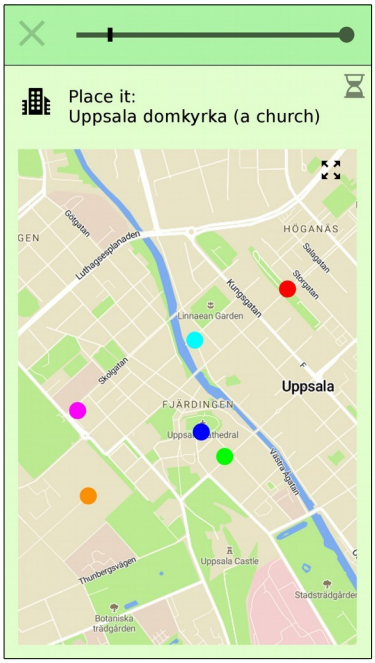

Figure 8: A Place-it question mock-up.

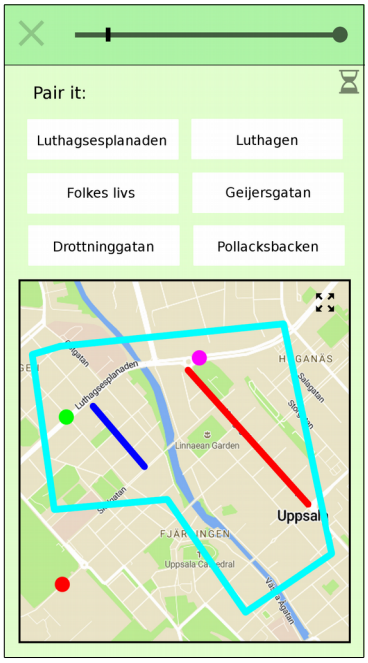

Figure 9: A Pair-it question mock-up.

4 Implementation

It was decided to implement Location Educator for the Android operating system for use by Android mobile phones and tablets, with focus on mobile phones. The programming language is Java for the logic and XML for layouts. The implementation happened in different steps, each of which is described in the following sections.

4.1 Data model

The first step involved defining in detail what data is needed by the application in order to comply with the requirements and support the necessary functionality. There are multiple entities, including *exercise*, *level*, *level-category*, *geographic object* and *question*, and relations between those. For all this data to be structured, manipulated and queried, a database was necessary. Therefor, an entity-relation model was created which can be seen in Figure 10. A user has multiple exercises, an exercise has multiple level-categories, a level-category has multiple levels and a level has multiple geographic objects. In addition, there is the *quiz entity* which is a singleton and describes an active quiz. A quiz has multiple questions, and each question has one geographic object.

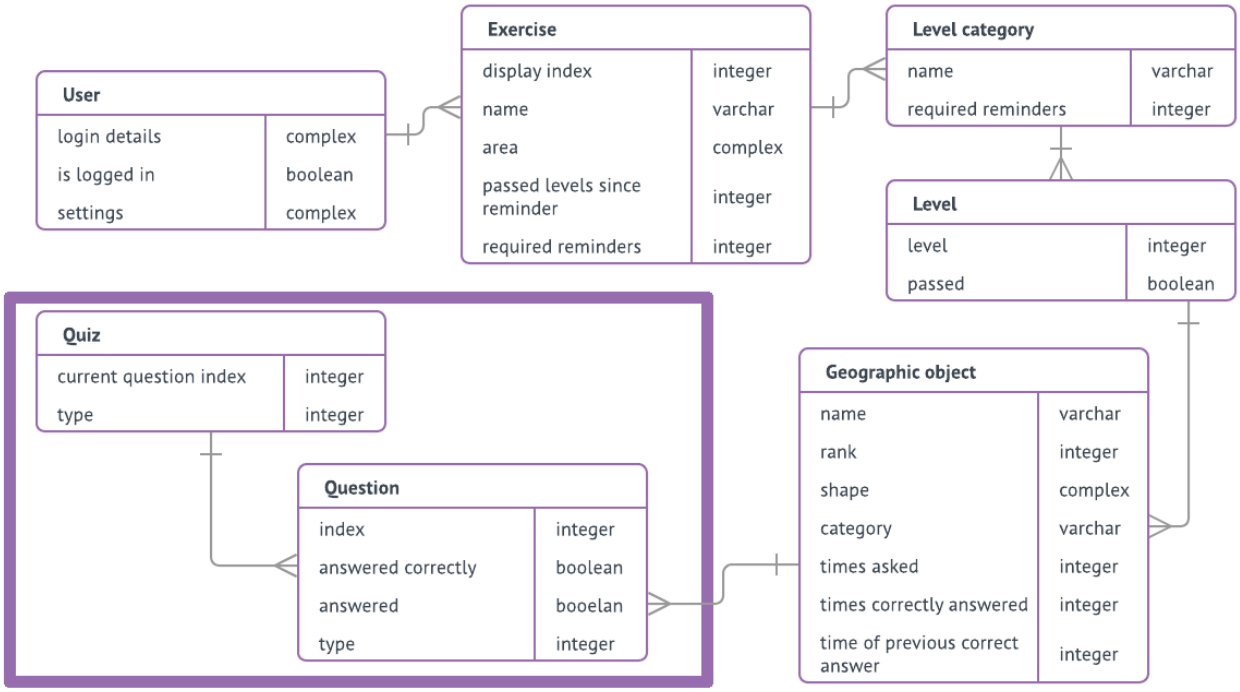

Figure 10: Entity-relation diagram for the permanent data of the application. In addition to stated attributes, each entity has a unique id. The one-to-many relationships are implemented by letting each child reference to the parent through an attribute containing the id of the parent. The purple box contains two entities that are constructed before a quiz. The others entities are more permanent.

The geographic object is the heart of the model. This entity has several attributes:

- **Category** – used during a question in a quiz, where the name along with the category is presented. Examples of values might be *public transport*, *education*, *religion*, *health*, *food*, *building*, *lake*, *waterfall*, *village*, *island and road*. Around 60 different categories exists. This is a more fine-grained category than the level-category (which is one of *settlement*, *road*, *nature*, *transport* and *construction*).
- **Shape** – defines the shape and location of the object through a set of coordinates. It might be a single node, a line, or a closed line.
- **Rank** – used to divide objects into levels inside a level-category. High rank means the object is of considerable importance and should be placed in an early level, and therefor be presented to the user at an early stage in the exercise progression.
- **Times asked**, **Times correctly answered** and **Time of previous correct answer** – is stats of answers to questions asked about this object. These stats are useful when selecting objects for reminder-quizzes, when determining number of required reminder-quizzes and when selecting geographic objects for a Pair-it question. Stats are of half significance in follow-up quizzes.

Also worth mentioning is the (number of) **Required reminders** attribute of Exercise and Level-category entities. These numbers are semi-randomly updated after each quiz, influenced by the number of problematic objects. A problematic object is an object with a bad success rate, or hasn't been correctly answered in questions for a long time. When these numbers are larger than zero, the level-progression is halted until required reminder-quizzes are completed.

This data-model was implemented using an *SQLite database*. An SQLite engine is available on android devices natively, and useful to store complex data locally on a device. [14]

4.2 Fetching data

Once established what data is needed, it was time to implement the gathering functionality of this data. Most of the data is related to geographic objects. Since the exercise area is arbitrarily specified by the user, and since it's impossible to store detailed geographic data about the whole world in the executable, this data must come from online sources.

One alternative was to create a server which had detailed geographic data stored locally, with world coverage. The server would supply the clients with requested data that would suit the clients' need perfectly. This idea was put aside because of problems that may arise related to server management. Instead, the goal was to let the application download data from public geographic data providers and process this data locally.

It was decided to use geographic data from *OpenStreetMap* (abbreviated OSM). OSM is a collaborative project for creating maps of the world, and has extensive coverage which is updated by an active community. OSM-data is free for use by anyone and a number of different APIs[1] exist for extracting the data. [15]

OSM-data consists of elements, which are the basic components of OpenStreetMaps's conceptual model of the physical world. An element is one of three things: Node (a single point in the world defined by longitude and latitude), Way (a collection of nodes in a path) or Relation (a collection of nodes, ways or other relations). In additions to coordinates, an element may have one or many tags. A tag is simply a key and a value; two arbitrary strings. There is no fixed dictionary of of tags but there exists conventions how they are supposed to be used. Since OpenStreetMap is a collaborative project (meaning the data can be added and modified by anyone), the quality and relevance of tagging varies. Some examples of tags are *name=Uppsala* and *place=City*. [15]

Location Educator needs small portions of the OSM-data, specifically geographic elements inside a size-limited polygon. For this purpose, the *Overpass API* service is well suited. This is an API that is optimized for providing OSM-data for a small part of the world. A query is sent to the Overpass API using the *Overpass QL* query language and the dataset that corresponds to this query is returned. [16]

Using the Overpass QL language, a query was constructed for fetching every element with a name-tag inside a polygon defined by a set of coordinates. Using some special features of the Overpass API it was possible to make Overpass perform some data processing on the raw OSM-data before

1 *API* is an abbreviation for Application Programming Interface.

the data was delivered. Through this, it was possible to receive only the *hull* (an encapsulating path) around complex objects instead of the whole objects. This shrunk the size of the data downloads considerably and decreased the processing time on the client side.

4.3 Processing data

After the data has been fetched from the Overpass server, it needs to be processed into entities of the data-model. This involves several steps.

4.3.1 Incoming data → build instructions

The incoming data arrives as line by line of xml-data [16]. A number of lines together describe an OSM-element. It is not preferable to download all data in a big chunk since that might cause the memory to run out on resource-limited mobiles. Therefore, this was implemented in a way that very little data needs to be in main memory at the same time. An incoming line of xml-data is scanned with regular expressions in order to extract the information in the line; is it a beginning of a new element, a part of the geographic shape of an element, or tags describing an element? This information is collected in a set of build instructions – instructions for building a single geographic object of the data-model. When a line of xml-data contains the beginning of a new geographic element, the set of build instructions currently collected is complete and used to build a geographic object. This construction process is described below. When construction of a geographic object is done, it’s stored in the application database and no longer needs to be stored in main memory.

4.3.2 Build instructions → geographic object

The gathered build instructions for building a geographic object contains coordinates describing the shape of an OSM-element along with a set of tags describing the element. Only elements with a name-tag are fetched from the Overpass server. Therefore, it’s guaranteed that one of the tags in the tag-set is a name but apart from that, the set of tags can be anything.

For the construction of a geographic object to be possible, a name along with a *super-category* and a *sub-category* is necessary. The super-category corresponds to the broad level-category and the sub-category is the more precise category for a particular geographic object. A closed set of super-categories as well as sub-categories was defined, and it is necessary to assign one category from each set to each geographic object during construction. Therefor, the set of tags in the build instructions needs to be converted into two categories. If this conversion fails, the tags aren’t satisfactory and the construction of this particular geographic object isn’t successful.

The conversion from tags into a super- and sub-category isn’t a trivial task. The problem is varying quality of the tags. Conventions exists of how tagging should be done in OSM, but there are no guarantees and you never know what tags will appear [15]. Also, an element usually has multiple tags; how to decide which one to pay attention to?

This was solved by creating a look-up table, listing all tags of interest and defining a super- and sub-category to each. The order of the list defines the importance of a tag, where low index means high importance. This order was roughly estimated. For each tag in the build instructions, the

corresponding entry is found in the look-up table, and the super- and sub-category of the tag with lowest index in the table becomes the categories of the geographic object.

In order to create this look-up table for conversion from tags to categories, it was necessary to examine tagging patterns in the OSM-data. For this, *TagInfo*[17] was used. This tool analyzes all OSM-data regularly, and reveals among other things rankings of tag-popularity and relations between tags.

In addition to a name, a shape, a super-category and a sub-category, a geographic object needs a ranking. This is necessary in order to present the user with *important* (in the sense important to learn its name and location) geographic objects early in the exercise progression, and move on to less important objects later. It's not enough to only use an object's size for this; there may be a huge old warehouse that is of much less interest than a compact town hall. It's not possible to extract a very accurate importance ranking using OSM-data alone. One potential solution would be to look at statistics from an OSM-element's Wikipedia-page[2] which may provide accurate importance, but it wouldn't be feasible to query for this information for each of a significant amount of geographic object. This was instead solved by extracting an importance-ranking from an OSM-element's number of edits in its version-history, the number of tags present, and also the element's size (where longer/ larger means higher importance). This importance-ranking method is questionable but does provides some relevance, and is achieved using OSM-data alone.

4.3.1 Final processing of geographic objects

When all incoming OSM-data has been attended there probably exists multiple new geographic objects in the application database. Once all these objects are present, there is need for some final processing of these objects; some of them need to be merged. The OSM-data contains many elements that aren't complete by themselves, for instance road segments. These are named elements which lie close to another element with same name. This isn't acceptable for the topical purpose; the answer to a question in a quiz may have multiple correct answers, which really isn't preferable. Therefore, merging of such objects is attempted.

All object with same names are extracted as groups. Merge is attempted for objects in each group. A merge between two objects is possible if the objects are in relatively close proximity. These two objects now become one object containing two shapes.

This is a vital step but it takes a long time to complete if there are a lot of geographic object. One reason for this is that it's not preferable to keep all objects in main memory at the same time – the device might run out of memory (a big exercise-area might contain tens of thousands of objects, and an object may contain hundreds of nodes). Therefor, objects need to be loaded from the application database repeatedly.

4.3.2 Geographic objects → complete exercise data

When above steps are complete, there are almost complete geographic objects in the database. These objects are all connected to the exercise currently under construction. The quiz-id attribute of

2 Many OSM-elements have a particular web-page in a *Wikipedia* database.

the geographic objects are at this point is set to minus one. Now, the complete structure described in the data-model is constructed from the geographic objects as follows:

- Geographic objects are grouped into levels based on super-category and ranking.
- Levels are grouped into level-categories based on the super-category of the objects.
- The level-categories are assigned to a newly constructed exercise.
- The exercise is assigned to a user.

4.4 Common operation

The most complex operation performed by the application is the construction of a new exercise. Once an exercise is constructed, the application basically presents the data in the database in a relevant way, defined by the application design and requirements.

When a quiz is initiated by the user, a corresponding quiz object is constructed and placed in the database. The database only holds one quiz object at the time, which is removed from the database when a quiz is completed. If a user leaves the application during a quiz, the quiz object stays in the database and when the user returns, the quiz is continued.

During a quiz, stats are gathered from the answers about the geographic objects of the questions. These stats are later used when constructing questions for reminder quizzes, as well as during construction of Pair-it questions. When a quiz is completed by the user, the number of required reminder-quizzes (described in the application design) is determined. This number is semi-randomized and also depends on the geographic objects' stats.

4.5 Integrated maps

During user's specification of a new exercise, during exploration and during quiz-exercises there is need for an integrated map on the screen. This was implemented using *Mapbox Maps SDK*[3] *for Android*, which provides highly customizable integrated maps for the Android platform [18]. The map is customized using a *Mapbox style document.*

- **Exercise area specification** in the New Exercise-activity requires an integrated map with functionality to draw a path. The user is presented with a map that a shape can be drawn on top of, provided the map is zoomed in to a certain zoom level (as defined by the application design). When the user taps the zoomed-in map, a marker is placed on the map at this location. Placed markers are connected with segments. Once three markers are placed, the shape automatically closes with the final segment. More markers can be added to further refine the shape; any number of markers is allowed. The user may clear added markers by tapping a button. Intersecting segments are detected using linear algebra and the *Create exercise button* is disabled. The map presented to the user contains text-labels; text in the map for the names of the various places. The Mapbox Maps SDK allows markers of custom

3 *SDK* is an abbreviation for Software Development Kit.

appearance to be placed anywhere on the map. The SDK also provides *Polylines* for representing segments between points.

- **Exploration** requires an integrated map containing representations of geographic objects. Objects are represented with map annotations; markers and polylines. Polylines are used for both open ended geographic objects (like roads) and objects with a closed shape (like areas of any sort). The SDK provides tapping functionality for these map annotations. When a user taps an annotation, the corresponding object's name and category is presented. The underlying map doesn't contain any text-labels. These labels would provide the user with the names of the geographic objects and would contradict the purpose of the exercise. The hiding of text-labels is specified in the Mapbox style document.
- **Quizzes** require integrated maps containing geographic objects, similarly to exploration. This map also doesn't contain any text-labels. Depending on the type of question, geographic objects can be tapped. If a correct answer is provided, the appearance of the object's representation in the map changes to indicate this, which is done by enlarging markers and thickening polylines. The color of the geographic objects are initially gray but when tapped turns into the particular color of this object.

4.6 Development tools

For development of the application, Android Studio was used. This is an integrated development environment specifically designed for Android development and provides much functionality that simplifies the production process. Among other things, it provides a layout editor where the developer can create layouts and corresponding XML code using a graphical interface. [14]

5 Evaluation

The implementation of the application resulted in an application that corresponds well with the functional and non-functional requirements, and the intended design. Some screenshots of the application can be seen in figures 11, 12 and 13.

5.1 Follow-up on initial user considerations

The finished product was tested by two people for an brief evaluation of the educational value, as a follow-up to the initial user considerations. Both were able to create an exercise for a desired area, start practicing and complete a quiz. The simplicity of selecting a category inside the exercise and press play to start a level was appreciated.

One of them had trouble answering before the provided time run out. The timer is only meant as a mild stressor and it might be preferable to adjust the provided time based on previous answering-times. It's however likely that users won't have this difficulty after gaining some experience.

Both users found the lightweight and relatively simple quizzes pleasant. This design enables users to complete a quiz without too much effort. Both found the application educational, but a more thorough analysis is necessary for a proper evaluation.

The users didn’t use the application for a long enough time to be able to provide opinions on the Reminder-system of the application.

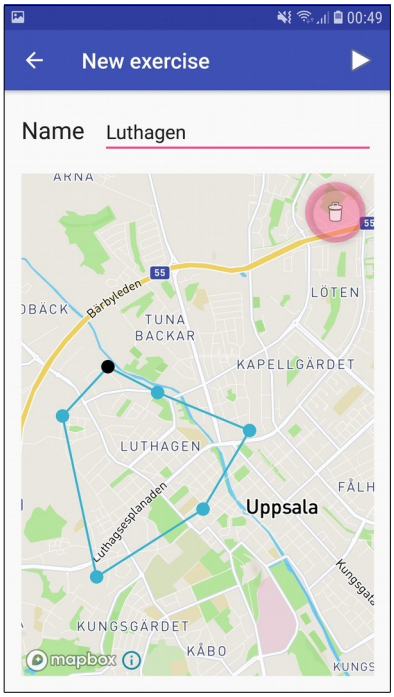

Figure 11: Screenshot of a New exercise activity.

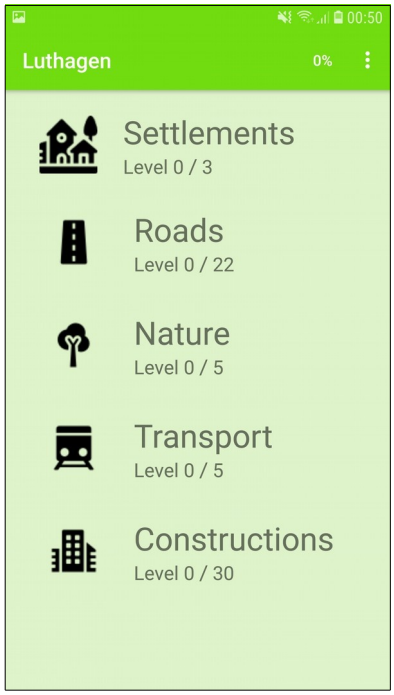

Figure 12: Screenshot of the main activity of an exercise called Luthagen.

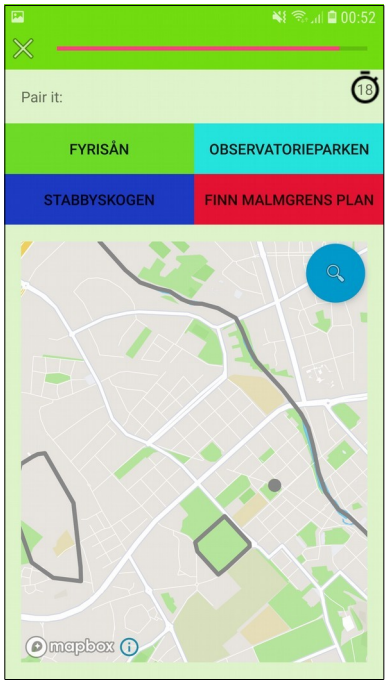

Figure 13: Screenshot of a Pair-it question during a quiz.

6 Current limitations and future work

The functional requirements declare functionality for predefined exercise areas, exercise sharing and progress sharing. None of these are implemented currently. These requirements exist mainly for some user’s potential desire of comparing progress with other users, and perhaps monitoring the progress of other users. This is likely important since some users may find the education more meaningful and motivational if it’s put in a social context.

Another limitation of the current implementation is the fact that when the application’s data is cleared, like for instance if the application is reinstalled, all exercises and progress is lost. An improvement would involve backing up data on a server, which might not be trivial since an exercise could contain a significant amount of data. This would also mean that a user can switch device and still keep the exercises and progress.

In the current implementation, the importance-ranking of geographic objects (used to determine a ranking among them for grouping them into levels) isn’t always accurate. A solution may involve looking at statistics from an OSM-element’s Wikipedia-page. It’s not possible for a device to query for this information for each geographic object since it would take too much time. A server would be necessary, where relevant data for the whole planet is stored locally. The server would then

supply the clients with the precise data a client needs. This could also provide a performance increase in exercise creation since the processing step currently performed on the device wouldn't be necessary. However, such an update involves much work and requires the need of one or many servers. The benefits seems relatively small compared to the effort.

Finally, the merging-step during creation of new geographic objects suffers from bad performance, which probably could be improved through optimization of the current implementation. The exercise construction may however take a long time, even with perfectly optimized code. Therefor, the progress of the construction should be presented, so the user realizes that he/she should do something else instead of waiting. A future potential improvement may involve letting a user start an exercise before it's constructed completely.

A more detailed evaluation of the application would be beneficial. The application could be tested by people in a group of diverse people of substantial size for a considerable amount of time. Then the members could be asked questions about the experience. What did they enjoy and what could be improved? They may also be tested on their knowledge of local geography before and after the use of the application in order to properly evaluate the educational value.

7 Discussion

The reason I wanted to develop this application was because I found myself looking for an existing application of this sort but couldn't find one. All I could find was applications for global and regional geography. I was using an Android application for language studies and realized the educational power of this. A friend of mine even learned to speak Swedish primarily through his tablet.

Another way to study local geography is by inspecting a map. This might not be ideal for everyone since some might not find it entertaining or very easy to find a way to preserve the names and locations in the memory. But there is a potential comfort in having a physical map to study; the map doesn't change or disappear. You can put it in a drawer and take it out whenever you want to refresh your knowledge. You can trust it in a way a mobile device can't be trusted. The mobile can break, one day it just won't function and all the data is lost. The current implementation of Location Educator unfortunately doesn't provide any support to battle this issue, like backing up exercise and progress data.

As mentioned in the Existing Application Analysis section, application developers are faced with crucial design decisions when developing quiz-based educational applications. There is often a balance between different benefits and potential hazards. The diversity in design of existing applications indicate the complexity of these design decisions. An extensive evaluation is required to quantify how successfully these aspects are handled and implemented in Location Educator. With that said, the initially determined design seems to have provided a useful application:

- The level progression functionality gives the user a goal and provides a supported exercise progression (important geographic objects are introduced before less important ones).

- The division of an exercise into categories (Settlements, Roads, Nature, Transport, Constructions) provides user with a seemingly comfortable level of freedom of choice.
- The three different types of questions (Name-it, Place-it, Pair-it) offer variation, gently guide the users towards correct answer, and are swift to answer.
- The educational support after an incorrect answer ensures that advantage is taken of the opportunity to offer guidance, by providing information about why the answer is incorrect.
- The counting down timer during a quiz-question offers a seemingly adequate level of stress.
- The Reminder-system ensures that new knowledge isn't forgotten.

I would like to become better at local geography in the city where I live, the areas in the archipelago of Stockholm where I spend the summers, and generally for places I'm traveling to. The names of local places tell a story about the areas' history. And the names help creating and preserving memories from these places. Since the application has been finished I have used it for studying the local geography of the city where I live. I really found the application useful; I learned local geography though it and I did find it entertaining.

References

[1] Clyde, L. A. (2004, 10). m-learning. Teacher Librarian, 32, p. 45-46.

[2] Seterra application for Android. https://play.google.com/store/apps/details?id=com.seterra

[3] World Geography application for Android. https://play.google.com/store/apps/details?id=com.age.wgg.appspot&hl=en

[4] Geo Sverige application for Android. https://play.google.com/store/apps/details?id=se.nimsys.geosverige.app

[5] Geo Challenge application for Android. https://play.google.com/store/apps/details?id=com.wetpalm.GeoChallenge

[6] Countries of the World application for Android. https://play.google.com/store/apps/details?id=com.socratica.mobile.countries

[7] MapPie application for Android. https://play.google.com/store/apps/details?id=com.learnpie.mappie

[8] Blank Map Quiz application for Android. https://play.google.com/store/apps/details?id=com.thatsverynice.mapquiz.blankmapquiz

[9] Duolingo application for Android. https://play.google.com/store/apps/details?id=com.duolingo

[10] McLeod, S. A. (2007). Stages of memory - encoding storage and retrieval.

[11] Ebbinghaus, H. (1885). Memory: A Contribution to Experimental Psychology, Dover, New York, NY.

[12] David M. Diamond, Adam M. Campbell, Collin R. Park, Joshua Halonen, and Phillip R. Zoladz (2007).The Temporal Dynamics Model of Emotional Memory Processing: A Synthesis on the Neurobiological Basis of Stress-Induced Amnesia, Flashbulb and Traumatic Memories, and the Yerkes-Dodson Law. Neural Plasticity, vol. 2007, Article ID 60803.

[13] McLeod, S. A. (2012). Zone of proximal development.

[14] Brian Hardy, Bill Phillips. Android Programming, The big nerd ranch guide. 1th ed. Atlanta: Big Nerd Ranch, Inc; 2013

[15] Jonathan Bennett. OpenStreetMap. Birmingham: Packt Publishing; 2010

[16] Overpass API documentation. [accessed 4 September 2018]. https://wiki.openstreetmap.org/wiki/Overpass_API

[17] Taginfo general information. [accessed 4 September 2018]. https://taginfo.openstreetmap.org/about

[18] Mapbox Maps SDK for Android documentation. [accessed 4 September 2018]. https://www.mapbox.com/android-docs/maps/overview